traveling to Rome: a retrospective on the journey

john wilkes, hp laboratories

R2D2 workshop, Cambridge, UK
2008.05.12-13

contents

- why did we do it?
  - scene-setting; motivation; problem spec
- what did we do?
  - a set of descriptions
  - a set of tools => solutions
- what did we learn?
  - things that went well; things that didn’t; surprises

Why did we do it?

Goal: lights-out data center

Business needs

– predictability

– rapid, reliable responses to changing demands

Why did we do it?

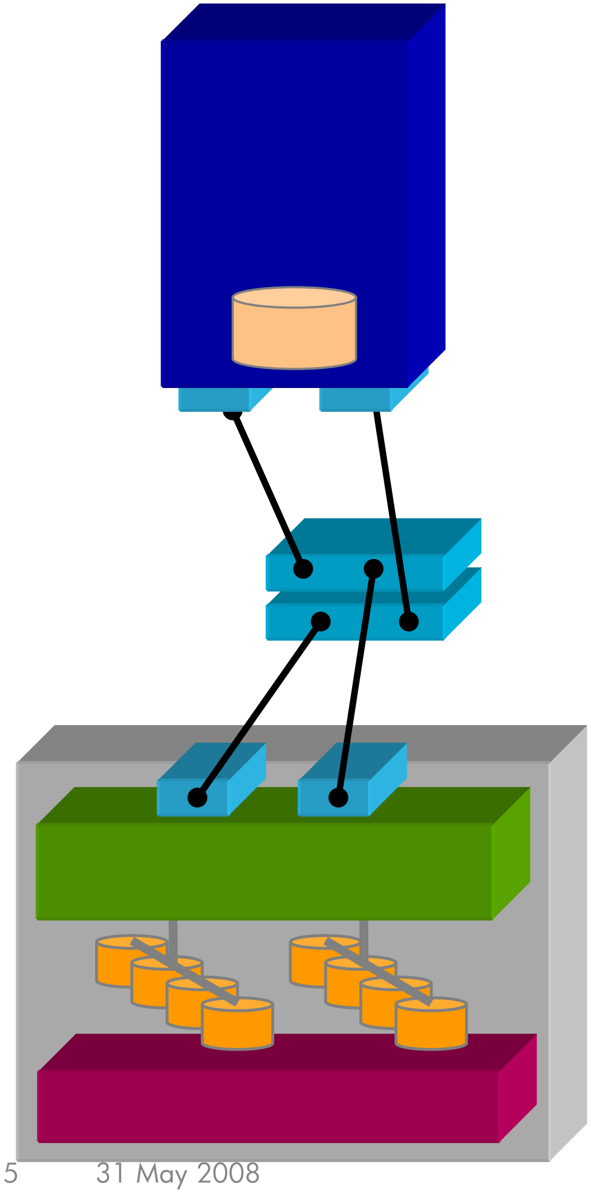
Complexity: too many storage management tasks

1 Activate licensed features in fabric elements
2 Add SAN resource domain (fabric + devices) to existing installation
3 Add host to existing FC fabric
4 Add hub to existing FC loop/fabric
5 Add peripheral disk device to bridge
6 Add peripheral disk device to storage array
7 Add port to storage array
8 Add switch to existing FC fabric
9 Add tape drive or library to bridge
10 Analyze SAN topology for single points of failure
11 Analyze SAN topology for traffic hot spots
12 Analyze device behavior to predict failures
13 Assign IP addresses to SAN components
14 Assign OS to run in partition/on platform
15 Assign action for event response
16 Assign free volume to OS/application
17 Audit actual configuration against planned/intended config
18 Audit firmware configuration
19 Audit software configuration
20 Boot OS in partition/on platform
21 Change OS or OS FC driver revision
22 Change cabling to service/management modem(s)
23 Change cabling to service/management network hub
24 Change cabling to service/management serial hub
25 Change cabling to service/management server(s)
26 Change fabric cabling to HBA
27 Change fabric cabling to use spare port
28 Change fabric internal topology (ISL's)
29 Configure and compile OS kernel
30 Convert existing fabric to cascaded fabric
31 Convert existing fabric to fully redundant fabric
32 Convert host bus adapter from FC-SW to FC-AL or vice versa
33 Convert single-initiator SCSI bus to multi-initiator
34 Convert two existing fabrics into a single fabric
35 Diagnose I/O errors
36 Diagnose directed path/device I/O (online, offline)
37 Diagnose system crash/hang
38 Download FC host bus adaptor firmware
39 Download FC switch firmware
40 Download storage array firmware
41 Download tape library firmware
42 Failover broken host bus adapter
43 Failover broken intra-switch port or trunk (ISL)
44 Failover broken storage array port or link
45 Failover broken switch port or link
46 Find physical location of specific device or fabric element
47 Install new FC-AL loop
48 Install new FC-SW fabric
49 Install new host
50 Install service/management software (servers, agents)
51 Install software, patches, service packs
52 Install storage array (Shark, EMC, HDS, Clariion)
53 Install tape system with shared drives
54 Install tape system with unshared drives and shared robotics
55 Mount OS file systems
56 Online/offline FC-SCSI bridge
57 Online/offline OS volume manager objects (mirrored, concatenated, etc)
58 Online/offline host bus adapter
59 Online/offline intra-switch trunk (ISL)
60 Online/offline path in multipath-capable OS
61 Online/offline peripheral device
62 Rebuild system for disaster recovery
63 Replace FC-AL hub
64 Replace FC-SCSI bridge (SAN Data Gateway, NUMA-Q FC Bridge)
65 Replace FC-SW switch (single switch fabric, multiple switch fabric)
66 Replace SAN management server
67 Replace failed director/controller in storage array
68 Replace host bus adaptor
69 Replace host
70 Replace peripheral device
71 Replace platform management server
72 Replace tape library robotics
73 Reserve tape media and storage slots within tape library
74 Reset/power-cycle FC-SCSI bridge
75 Reset/power-cycle entire installation (power-fail, first bringup)
76 Reset/power-cycle host platform
77 Reset/power-cycle peripheral devices (on bridge)
78 Reset/power-cycle storage array
79 Run offline diagnostics (using idle/disused system components)
80 Run online diagnostics (using "active" system components)
81 Sanitize used fabric elements to safely reuse in new fabric (clear NVRAM)
82 Set/view "POST" diagnostic level
83 Set/view "business continuation volumes" (BCV)
84 Set/view OS configuration files/registry
85 Set/view OS volume manager volumes
86 Set/view SNMP trap destination
87 Set/view backup schedule
88 Set/view event reporting threshold
89 Set/view event-/error-report destination
90 Set/view online diagnostics error threshold trigger
91 Set/view phone-home/email-home destination
92 Set/view service/management authentication (passwords)
93 Set/view storage array LUN masking and LUN mapping
94 Set/view storage array volume definition
95 Set/view switch ISL topology
96 Set/view switch zoning
97 Set/view system boot parameters (device, flags, etc)
98 Set/view vital product data (diary RAM)
99 Test (acceptance) post-install/-repair
100 View/search system logs (OS, platform, fabric element, etc …

list from
Stuart Friedberg,
Veritas

Why did we do it?

Complexity: too many touch points



To add a block volume:

- logical volume manager
- storage-network interface cards

- storage network switches (zones)

- disk array ports (LUNs)
- logical unit (LU)
- physical volume usage

Why did we do it?

Complexity: performance

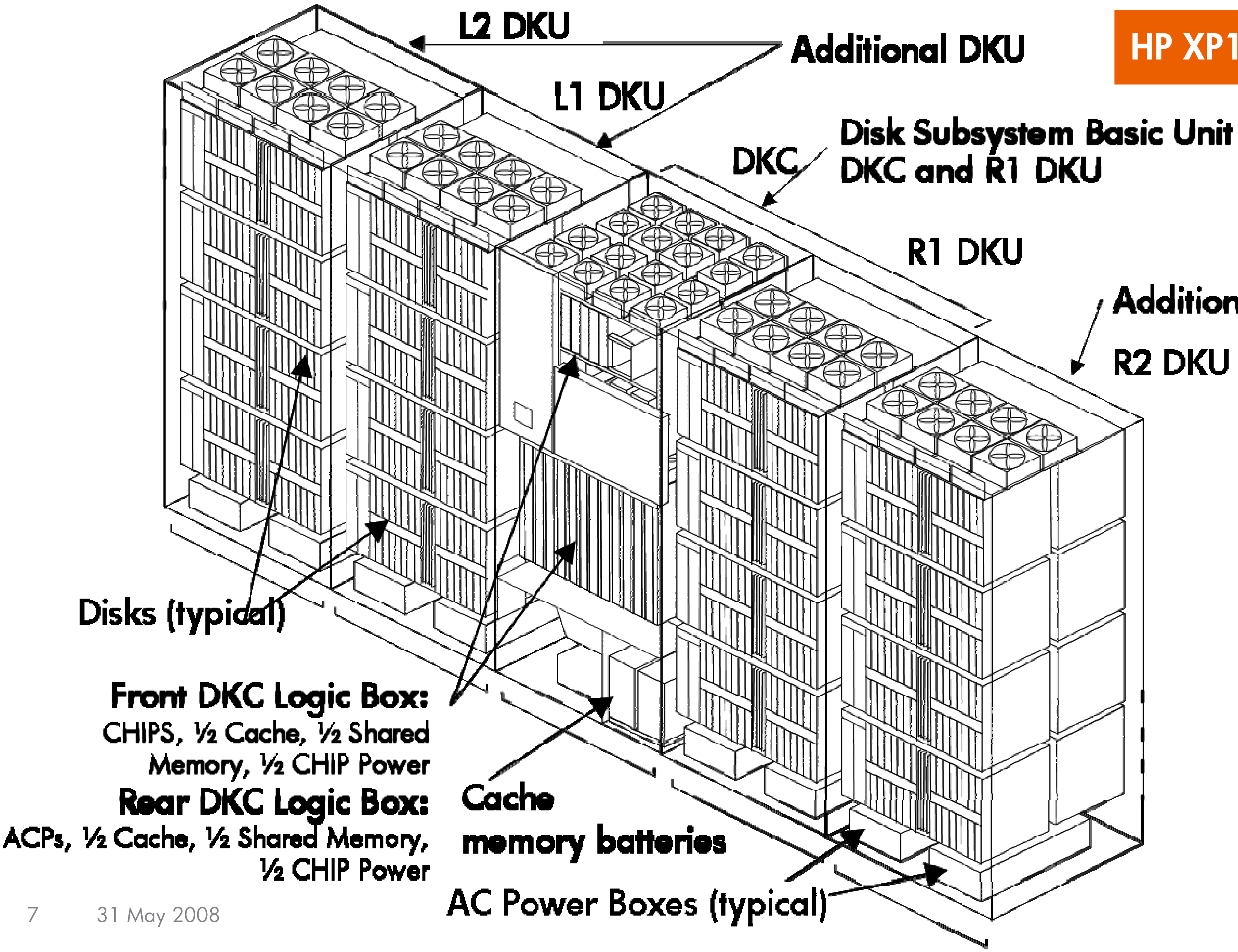
- Strong non-linear performance behavior
  - sequential vs random access
  - cache hits
  - multiple devices, paths
  - workloads are not additive

➔ 50–200x performance effects

- sequential I/O: 50MB/s
- random I/O: 0.1MB/s

Why did we do it?

Complexity: storage system structures



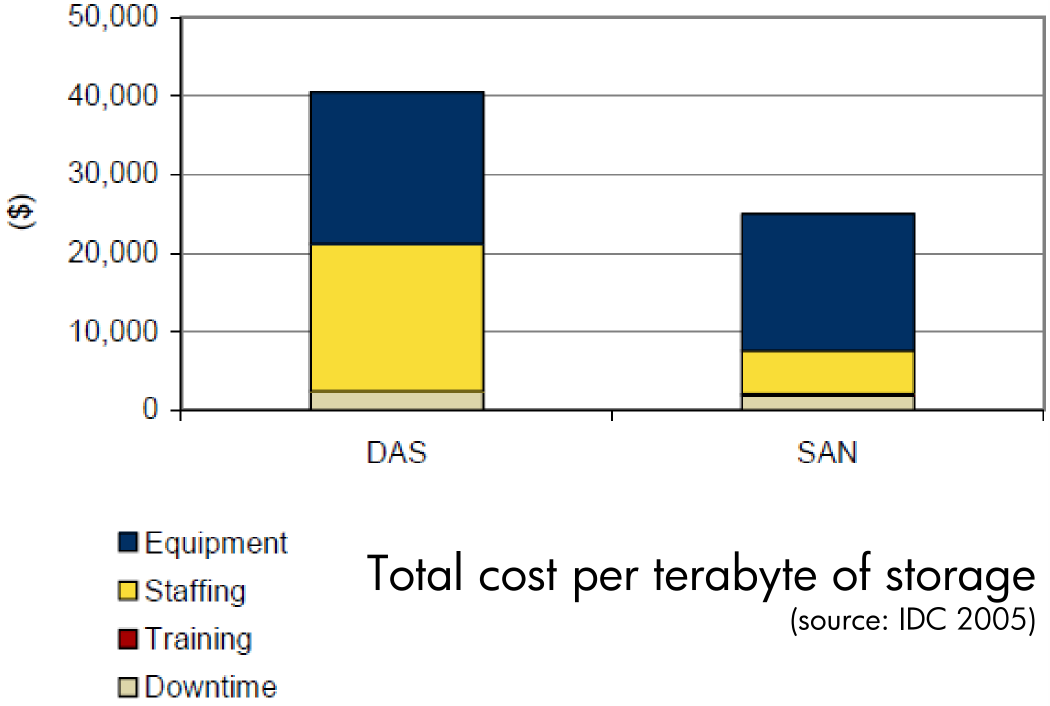
HP XP1024 disk array

Why did we do it?

People are getting more expensive

- Storage costs are dropping
  - 1995: ~$5000/GB raw
  - 2005: $0.5/GB raw

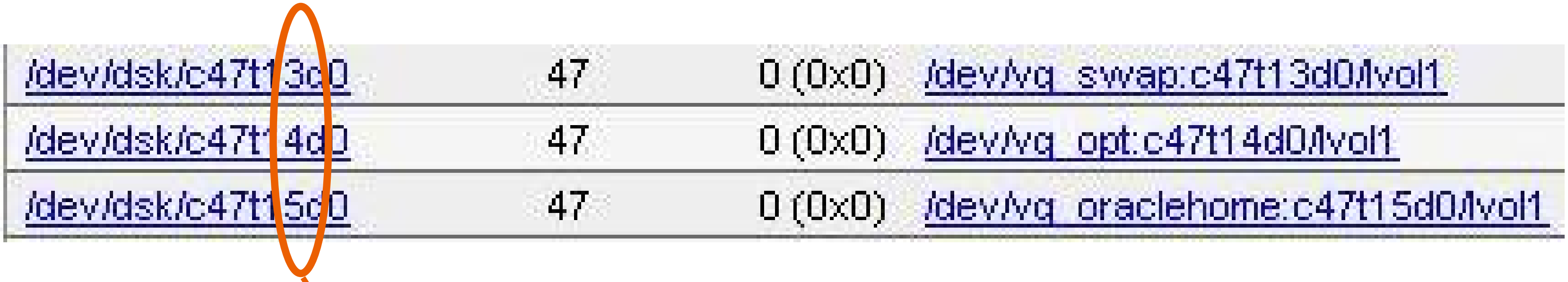
- Administrator costs are not
  - 2004–5 salary: $68k



Total cost per terabyte of storage
(source: IDC 2005)

Why did we do it?

Errors: many finicky details

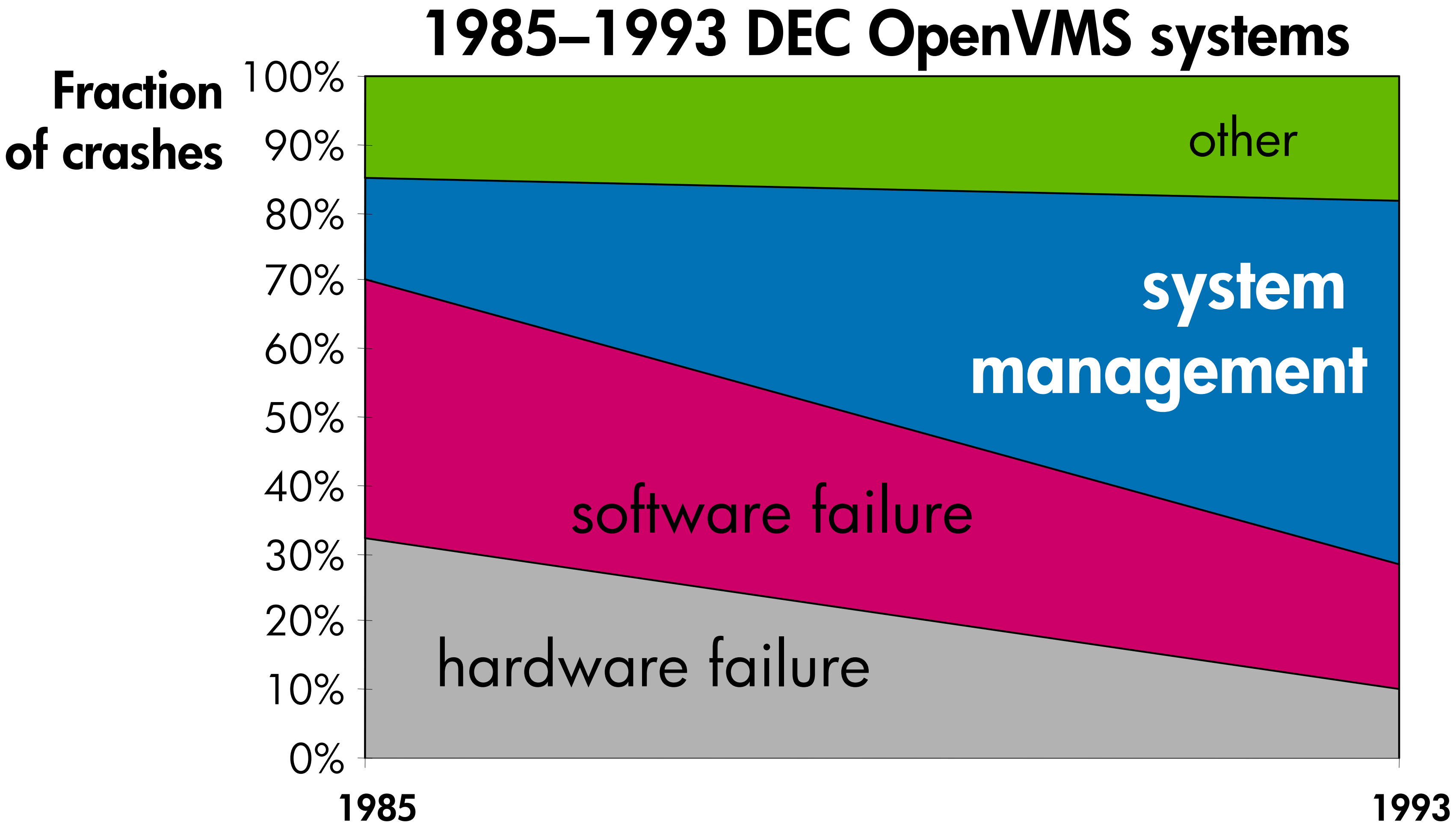


/dev/dsk/c47t13d0	47	0 (0x0)	/dev/vg_swap:c47t13d0/lvol1
/dev/dsk/c47t14d0	47	0 (0x0)	/dev/vg_opt:c47t14d0/lvol1
/dev/dsk/c47t15d0	47	0 (0x0)	/dev/vg_oraclehome:c47t15d0/lvol1

Transpose one digit, and
you wipe out the Oracle dbms!

Why did we do it?

Errors: humans are error prone



Brendan Murphy and Ted Gent, **Measuring System and Software Reliability using an Automated Data Collection Process**, *Quality and Reliability Engineering International*, **11**:341-353, 1995. © John Wiley & Sons.

G. Doré

Goal: the “lights out data center”

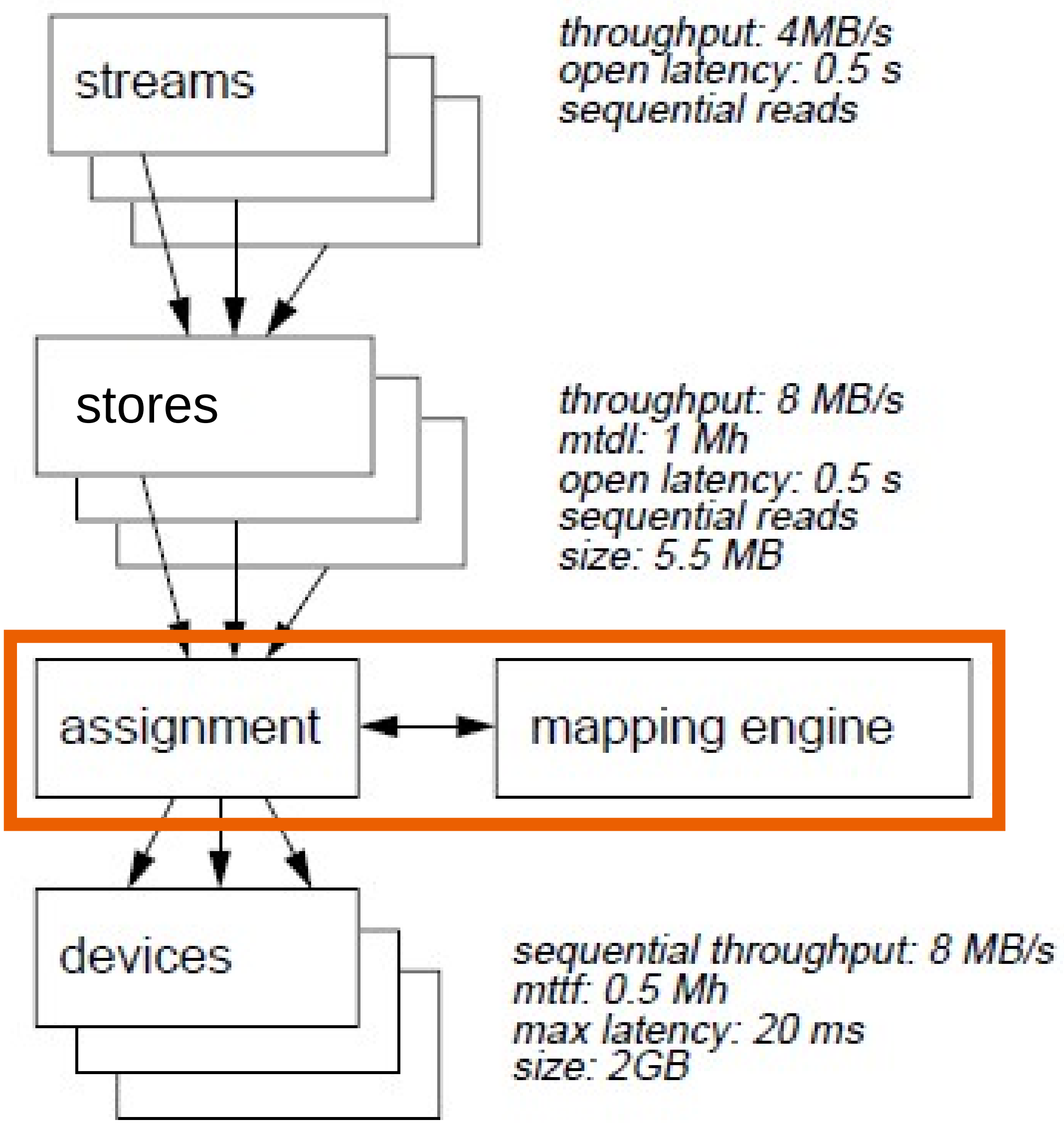
- Automate the design process
- Automate the configuration process
- Automate the system’s responses to changes

Tell us what you want …
not how to deliver it

what did we do?

What did we do?

Declarative specifications



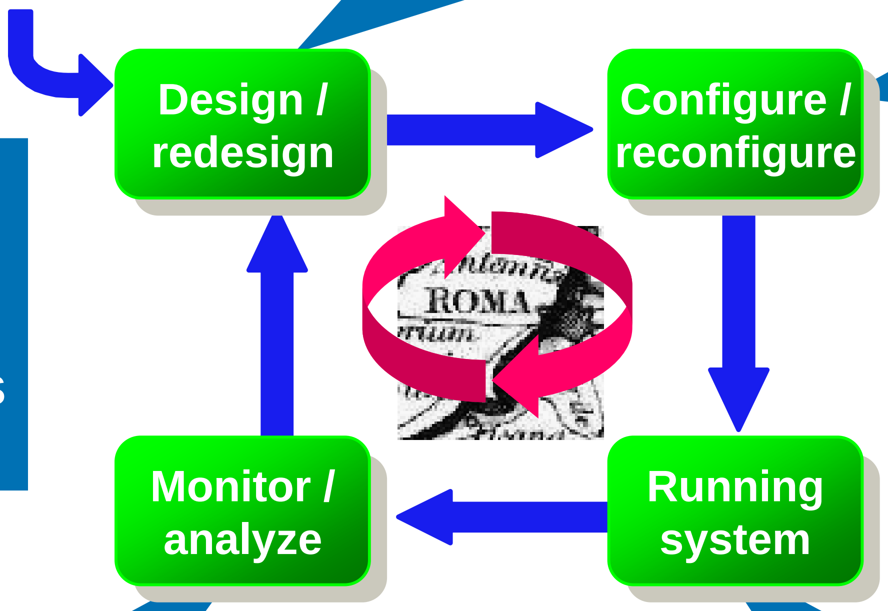
What did we do?

Overall structure

(Changing) business requirements

Determine solution
- select devices+configurations
- assign load

Understand needs
- offered load
- system components
- system goals



Construct solution
- configure targets
- migrate data

Model-based automation is the glue that holds all this together

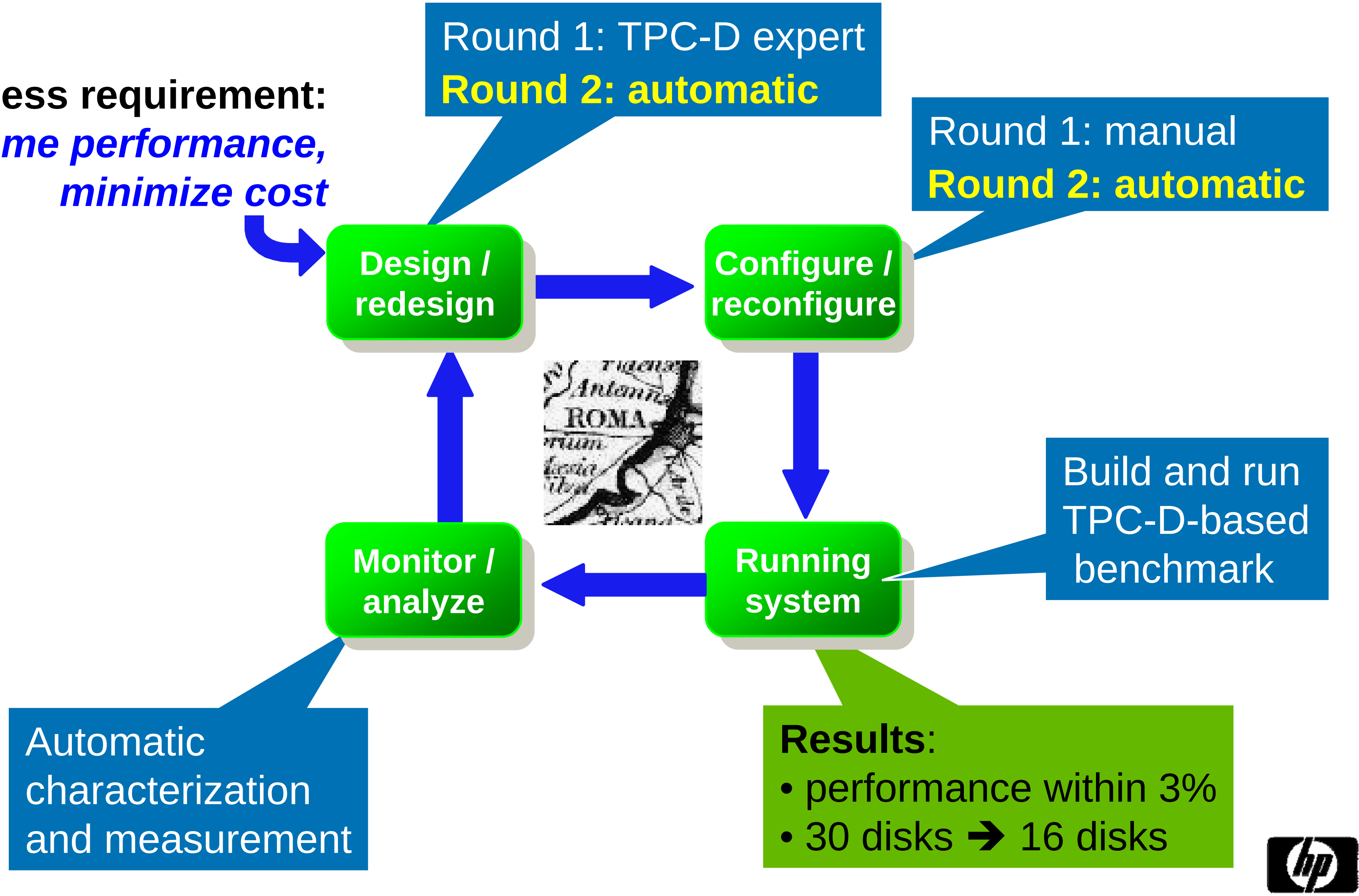
Monitor QoS
- offered load
- system response

Use the solution
- do work
- enforce QoS

What did we do?

TPC-D example (~1997)

Business requirement:
same performance,
minimize cost



Round 1: TPC-D expert
Round 2: automatic

Round 1: manual
Round 2: automatic

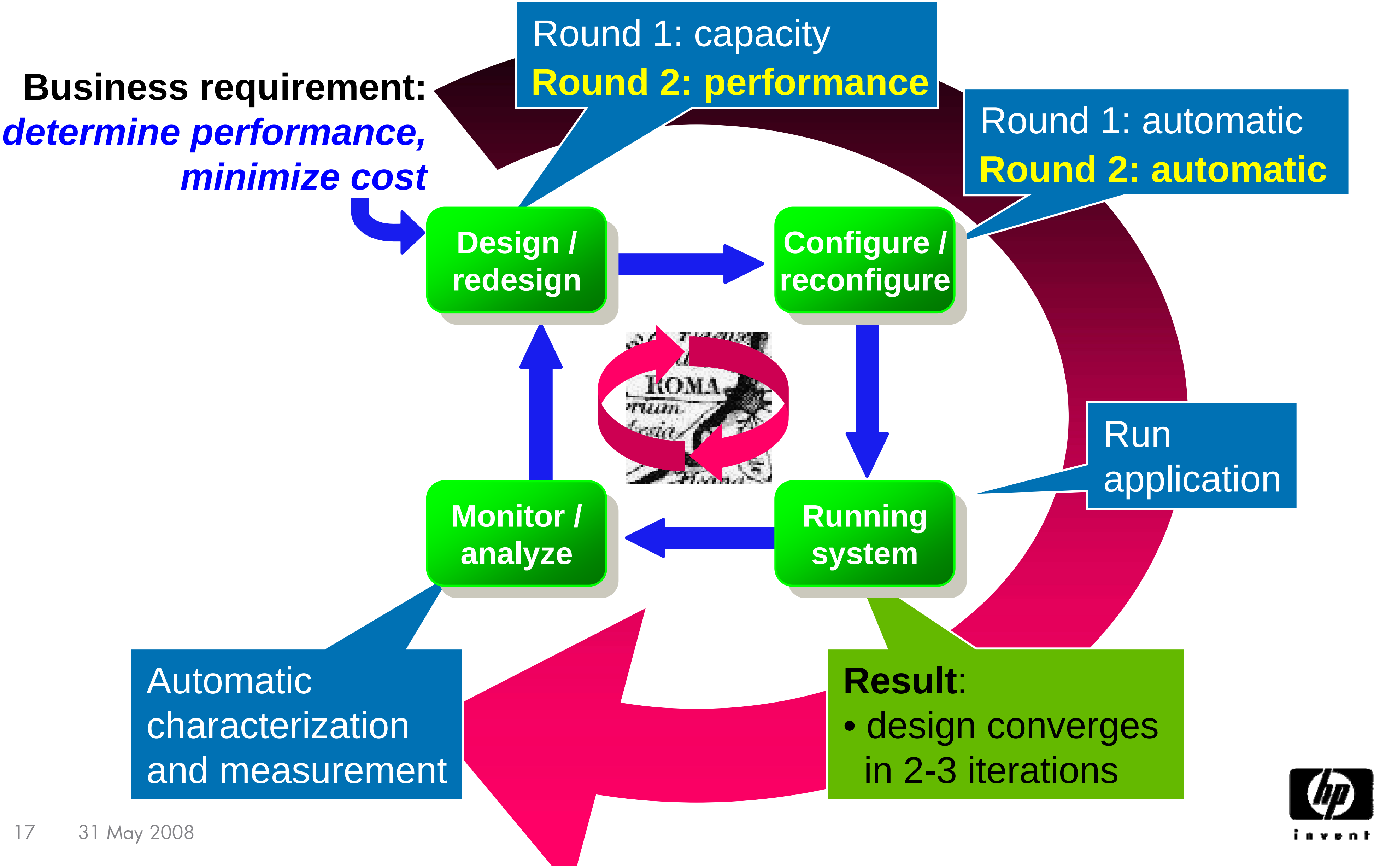
Build and run TPC-D-based benchmark

Automatic characterization and measurement

Results:
- performance within 3%
- 30 disks ➔ 16 disks

What did we do?

Hippodrome: closing the loop automatically (~2001)



Business requirement:
determine performance, minimize cost

Round 1: capacity
Round 2: performance

Round 1: automatic
Round 2: automatic

Design / redesign

Configure / reconfigure

Running system

Monitor / analyze

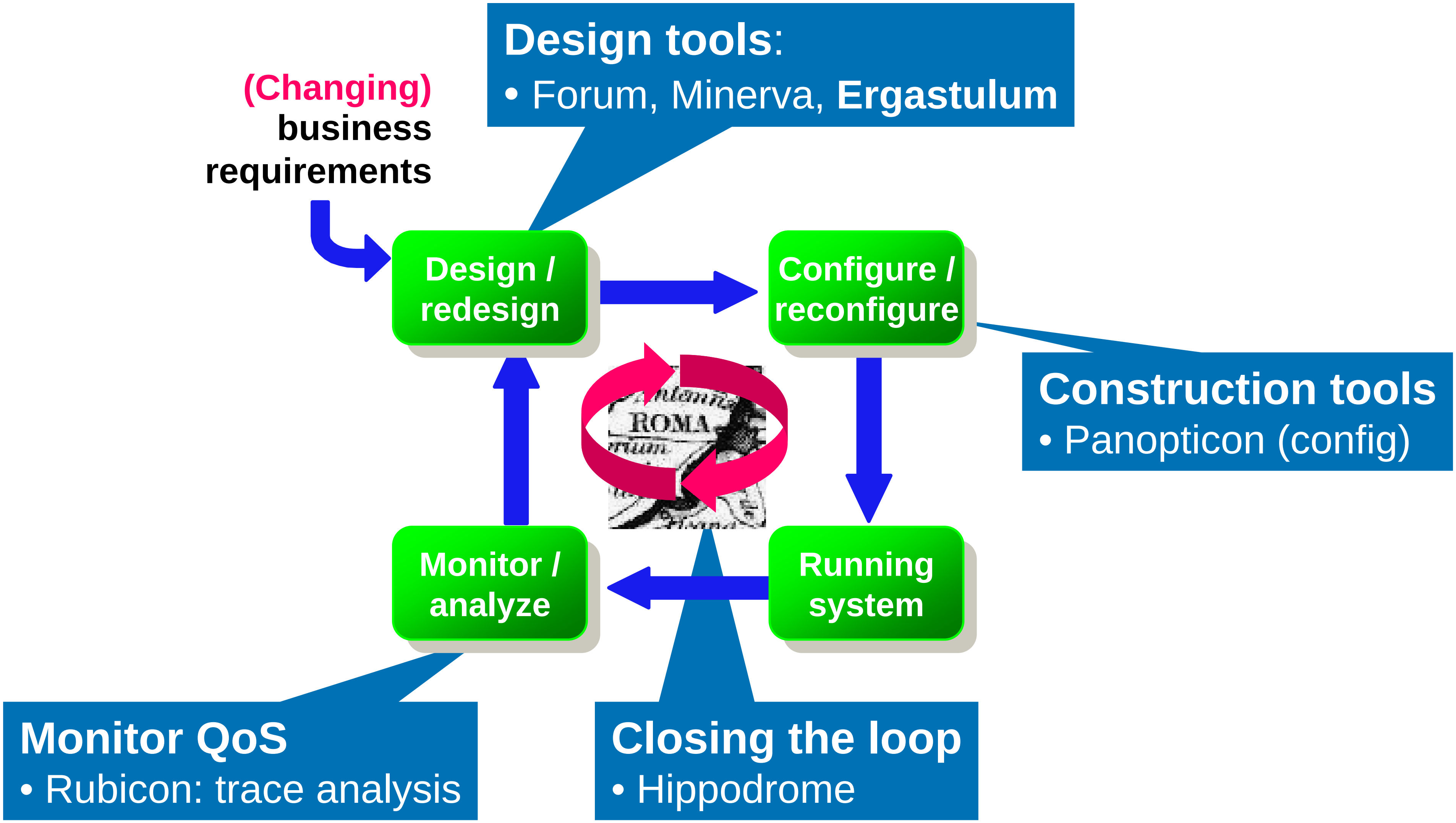
Run application

Automatic characterization and measurement

Result:
- design converges in 2-3 iterations

What did we do?

Tools



(Changing) business requirements

Design / redesign

Configure / reconfigure

Running system

Monitor / analyze

Design tools:

- Forum, Minerva, **Ergastulum**

Construction tools

- Panopticon (config)

Monitor QoS

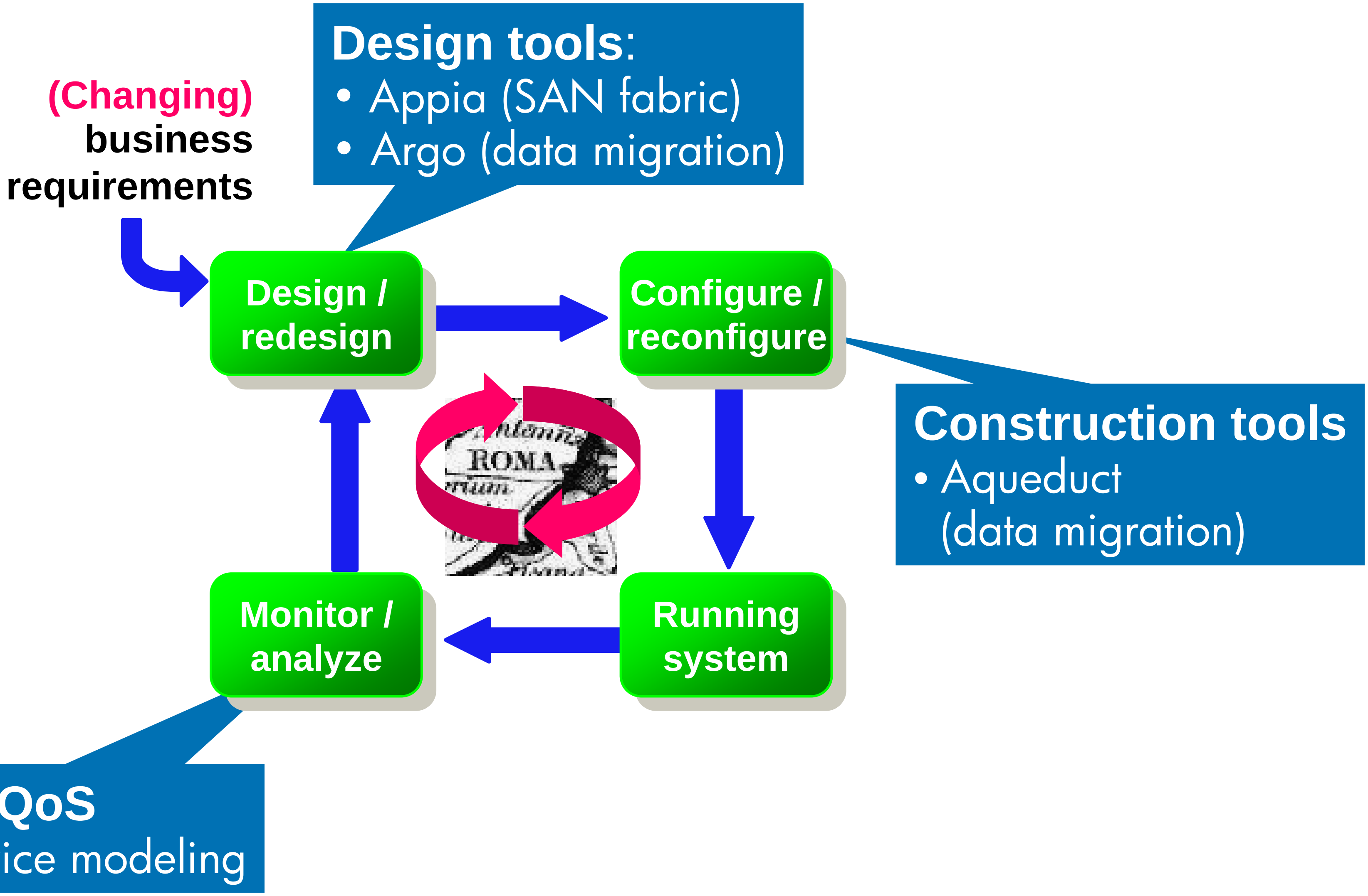
- Rubicon: trace analysis

Closing the loop

- Hippodrome

What did we do?

More tools



Design tools:
- Appia (SAN fabric)
- Argo (data migration)

(Changing) business requirements

Design / redesign

Configure / reconfigure

Running system

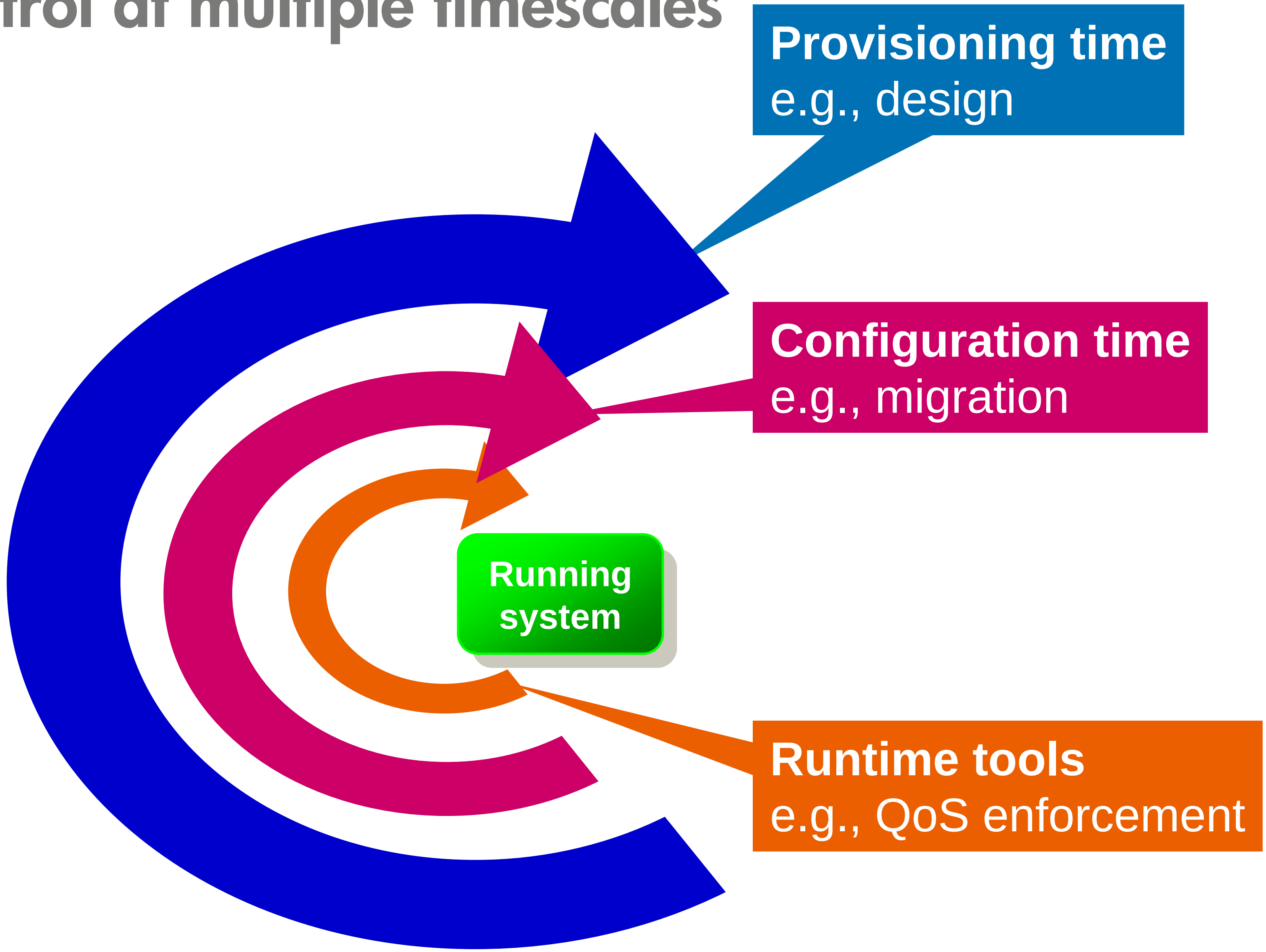
Monitor / analyze

Construction tools
- Aqueduct (data migration)

Monitor QoS
- Auto device modeling

What did we do?

Control at multiple timescales



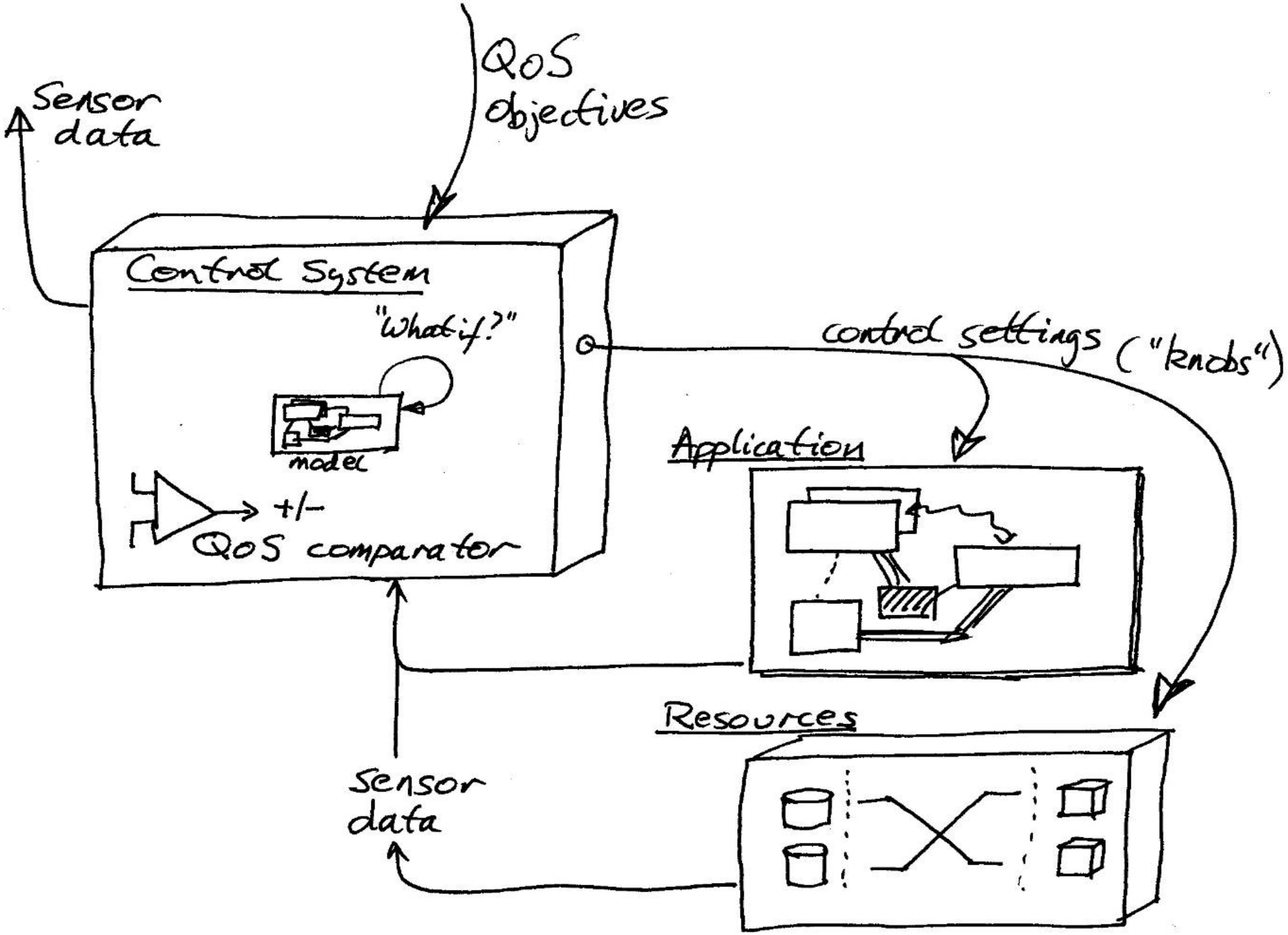
Provisioning time
e.g., design

Configuration time
e.g., migration

Runtime tools
e.g., QoS enforcement

What did we do?

Control loop



What did we do?

Rome: declarative specification language

- derived from Tcl [Ousterhout94]
- extensible
- used for inputs and outputs in tool pipeline
- multiple external representations
  - Latin: Tcl-like { curly braces }
  - Greek: XML < angle brackets >

```
store georgina {
  { capacity 100e9 }
  { boundTo  disk6 }
}
```

What did we do?

Eschew obfuscatory representations

- why say:

```
<sst:object type="diskDrive"
name="u"> <sst:object
type="serialNumber"> <cbt:string>1234-
5678</cbt:string> </sst:object>
</sst:object>
```

- when you could have said:

```
{diskDrive:u
   {serialNumber "1234-5678"}
 }
```

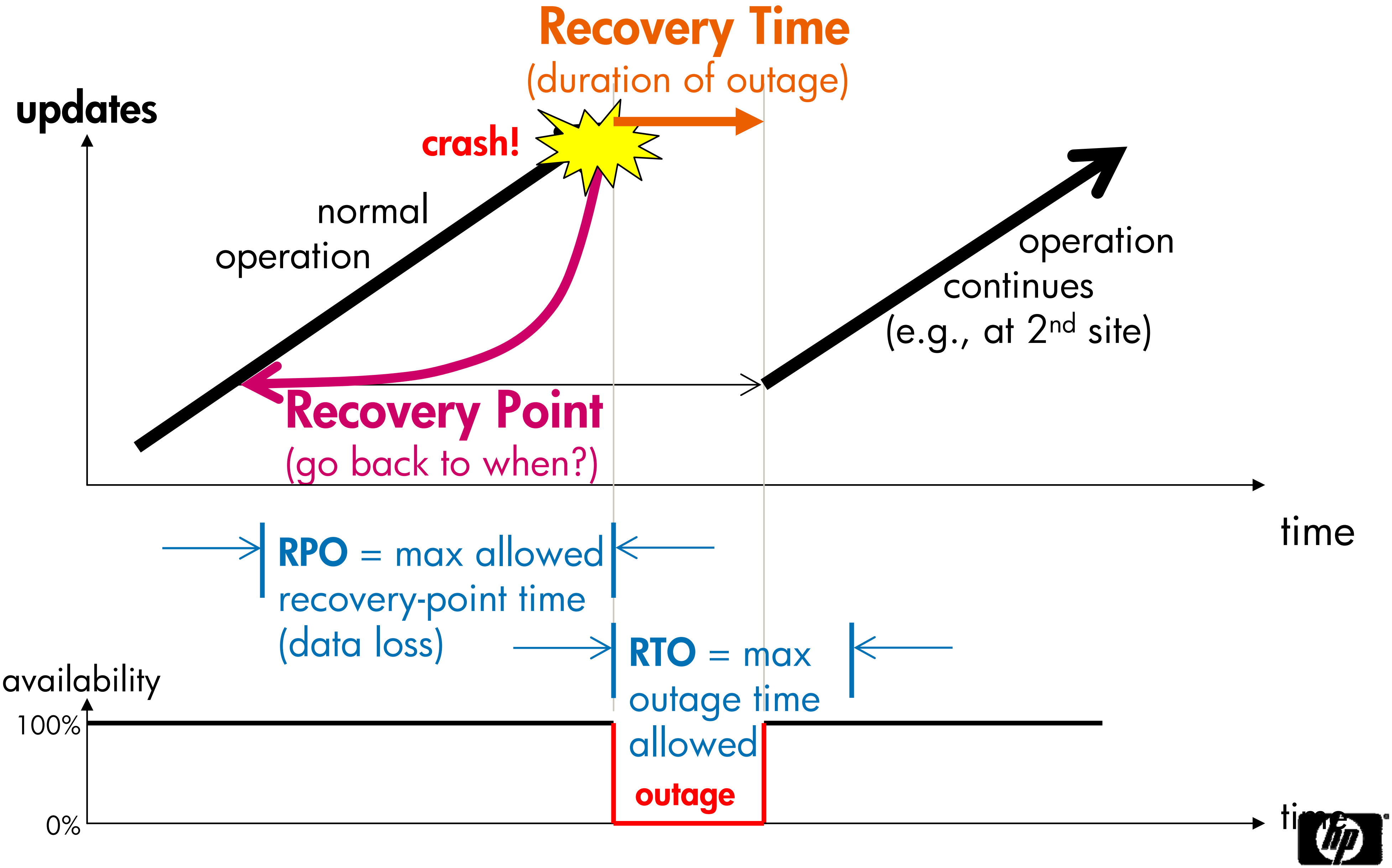
What did we do?
Business goals → SLA

- **QoS**
  - performance
  - capacity
  - cost
  - availability
  - reliability
  - security
- **QoI**
  - accuracy, completeness, relevance, believability, ...

an SLO

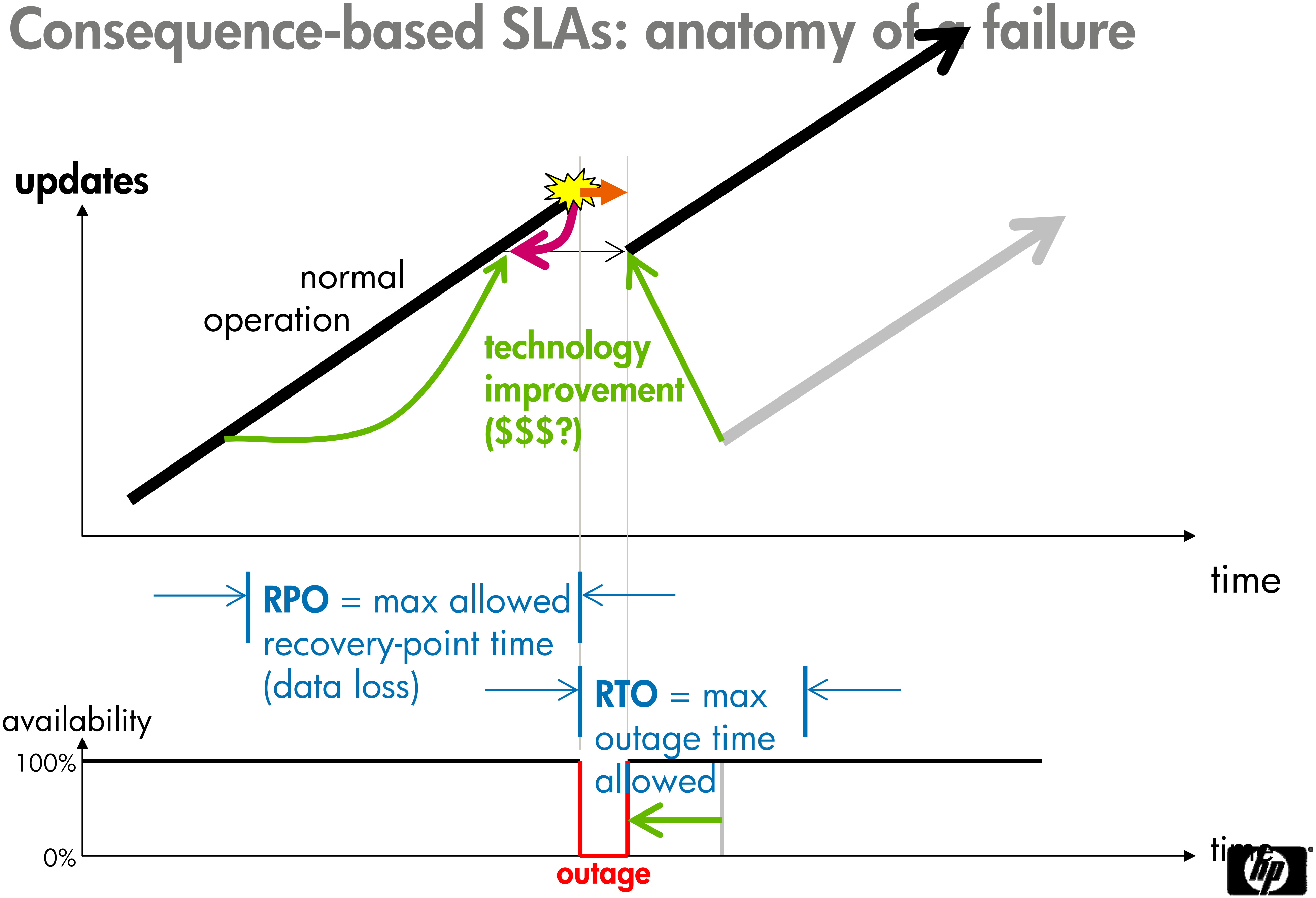
What did we do?

Consequence-based SLAs: anatomy of a failure



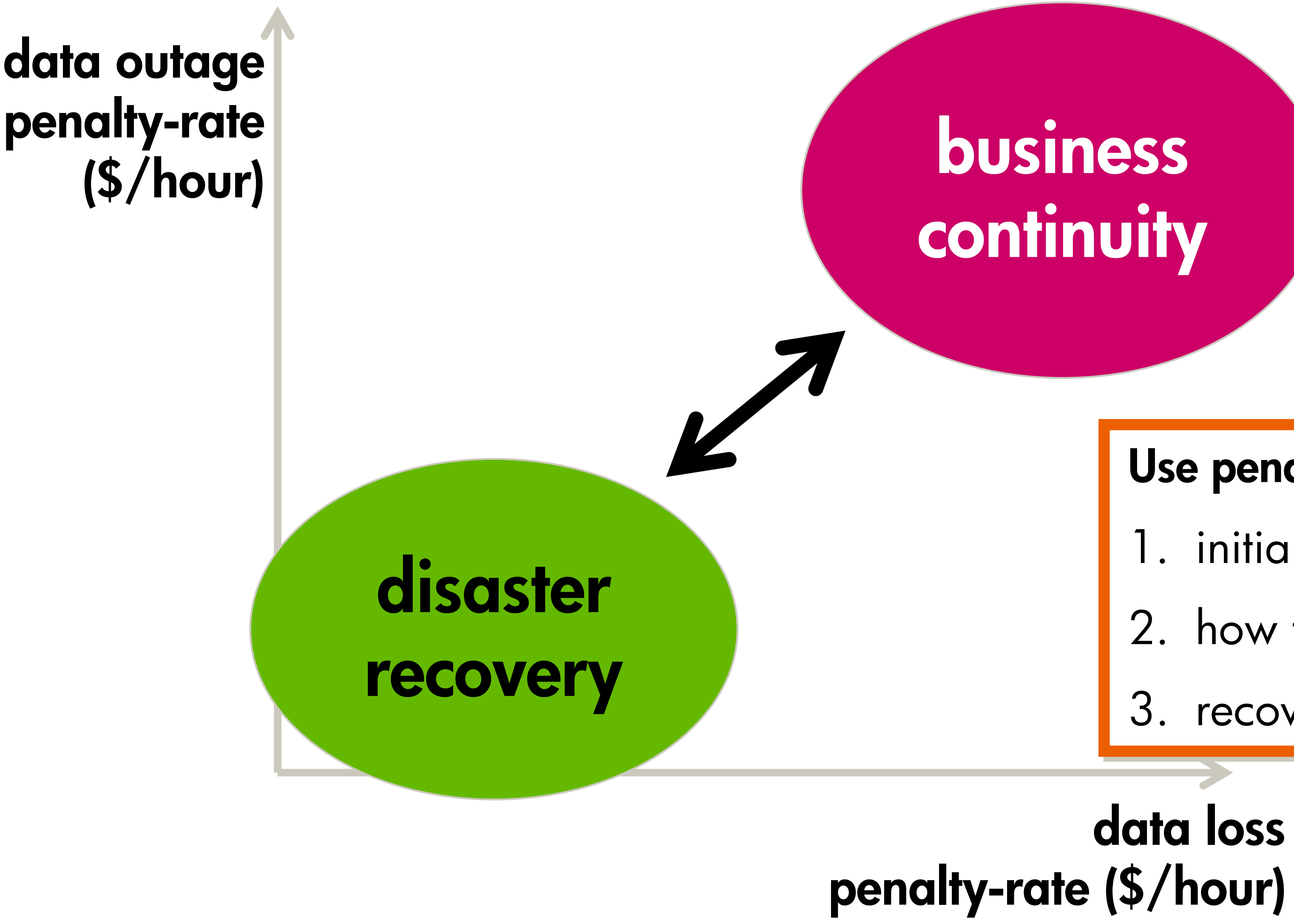
What did we do?

Consequence-based SLAs: anatomy of a failure



What did we do?

Consequence-based SLAs: failure goals


data outage penalty-rate ($/hour)
business continuity
disaster recovery
data loss penalty-rate ($/hour)

Use penalties to drive:

1. initial design
2. how to recover
3. recovery sequence

what did we learn?

What did we learn?

Trust matters

- Nobody will deploy a new system unless
  - they believe it will make their life better *and*
  - they believe it will not make their life worse
  - and sometimes …
    they have no choice

➔ Research topic: building trust

- how do we delegate?
- how do we limit the bad stuff?
- how do we persuade people?

What did we learn?

Simplicity matters

- Appia SAN designs often saved 2/3 cost
  - but customers wanted full crossbar-like designs

- People value:
  - symmetry
  - regularity
  - ease of understanding
  - ease of prediction
  - ease of adaptation

What did we learn?

Be clear what you are modeling

- Truth
  - reality: what's actually out there
- Beauty
  - goals: what you are trying to achieve
- Faith
  - measurements: what you think is out there
- Reason
  - predictions: what you think will be out there

What did we learn?

don’t be too early!

traveling to Rome: a retrospective on the journey

http://www.hpl.hp.com/research/ssp